Intelligent Interfaces

Single and Double Width

I/O Expansion Cards

User Guide

Intelligent Interfaces Ltd

Issue 4 18th January 1999

All correspondence should be addressed to:-

Technical Queries
Intelligent Interfaces Ltd
P O Box 80
Eastleigh
Hampshire
SO53 2YX

Tel:	023 8026 1514
Fax:	087 0052 1281
Email:	support@intint.demon.co.uk
URL:	http://www.intint.demon.co.uk/

Contents

Introduction

The Single Width I/O Expansion Card for any Acorn Risc OS-based computer with an expansion backplane provides the 1MHz Bus, User Port and Analogue to Digital Converter interfaces of the original BBC Microcomputer. Unlike the comparable Acorn Double Width I/O Expansion Card (which cannot be fitted in the A7000 or Risc PC), the single width card has a 25 pin Male D-type connector instead of a 20 way ribbon cable connector for the User Port (an adapter cable is available separately). Additionally, it cannot have a Musical Instruments Digital Interface (MIDI) upgrade fitted.

The Intelligent Interfaces ROM for the Single Width I/O Expansion Cards contains a new version of the I/O_Podule module which extends Risc OS and provides the same OS_Byte calls for accessing the 1MHz Bus, User Port and Analogue to Digital Converter as the BBC Microcomputer operating system. In addition, a number of new software interrupts (SWI's) have been introduced to simplify the use of interrupts from the User Port 65C22 Versatile Interface Adapter (VIA) in programs written in Assembler.

The original Acorn I/O_Podule module handled interrupts from the Analogue to Digital Converter by claiming the IrqV software vector and implemented the additional OS_Byte calls provided by the module by responding to the UKByte service call. The Risc OS 3.1 Programmers Reference Manual states that both these methods are deprecated. The Intelligent Interfaces I/O_Podule module claims the expansion card device vector to handle interrupts from the Analogue to Digital Converter and claims the ByteV software vector to implement the additional OS_Byte calls provided by the module. The new version of the I/O_Podule module increases the card's compatibility with other expansion and network interface cards.

Installation

The Intelligent Interfaces ROM may be supplied in one of two ways:

fitted in a new Single Width I/O Expansion Card

as an upgrade for an existing Single or Double Width I/O Expansion Card

Fitting a New Expansion Card

Before fitting the expansion card check that the following items in addition to this User Guide have been received:-

Single Width I/O Expansion Card

Half Width Rear Panel Blanking Plate, T-Piece and Screws (if supplied for pre A7000 and Risc PC computers)

If any item is missing contact the supplier.

Before the expansion card can be fitted in a 300 series computer an expansion backplane must be fitted.

Fitting the Expansion Card into Computers other than the A3000

1. Switch off the power to the computer.

2. Disconnect the computer from the mains supply.

3. Remove one of the blanking plates from the rear of the computer.

4. The Expansion Card and, if necessary,a half width blanking plate, can be fitted in any vacant position.

5. Reconnect the computer to the mains supply.

6. Switch on the power to the computer.

7. Press F12 and from the command prompt type

```
*RMReInit I/O_Podule
```

Fitting the Expansion Card into an A3000 Computer

The expansion card plugs into the external connector on the right hand side (viewed from the front) of the computer.

1. Switch off the power to the computer.

2. Disconnect the computer from the mains supply.

3. Plug the expansion card in its metal case into the external connector.

4. Reconnect the computer to the mains supply.

5. Switch on the power to the computer.

6. Press F12 and from the command prompt type

```
*RMReInit I/O_Podule
```

Fitting an Upgrade ROM in an Existing Expansion Card

1. Switch off the power to the computer.

2. Disconnect the computer from the mains supply.

3. Remove the existing expansion card from the computer.

4. Note the orientation of the existing ROM and carefully remove it using an IC extractor or a small screwdriver used with extreme caution.

5. Ensure that the upgrade ROM is correctly orientated and insert it carefully into the socket ensuring that all the legs locate properly into the holes and are not bent under on insertion.

6. Refit the expansion card into the computer.

7. Reconnect the computer to the mains supply.

8. Switch on the power to the computer.

Confirming that the Expansion card has been Installed Correctly

Type

```
*Podules
```

This should list the I/O Expansion Card as

```
Podule n: I/O Expansion Card
```

together with any other expansion cards fitted.

The I/O Expansion Card Software

The I/O_Podule module is contained in ROM on the Expansion Card and is loaded and initialised at power on or at a subsequent reset. To check that the module is present type

```
*Help I/O_Podule
```

The following (with possibly a later version number) should be displayed

```
==> Help on keyword I/O_Podule
Module is: I/O Podule      2.00 (27 Aug 1997)
```

I/O_Podule_Hardware (SWI &40500)

Purpose
To return the base address of the I/O Expansion Card.

Parameters
None

Results
R1 - base address of the I/O Expansion Card (base address of the ROM synchronous address space)

Example
```
   10 REM Determine base address of I/O Expansion Card
   20 SYS "I/O_Podule_Hardware" TO ,base_address%
   30 PRINT "Base address of I/O Expansion Card is ";~base_address%;"
hex"
   40 END
```

User Port 65C22 Versatile Interface Adapter (VIA) Interrupts

Five of the possible 65C22 VIA interrupt sources have been assigned device numbers

0	Shift Register
1	CB2 Input
2	CB1 Input
3	Timer 2
4	Timer 1

A device interrupt driver routine claimed, enabled, disabled and released using the following four I/O_Podule SWI' s is entered with

R0 - device number
R11 - base address of the 65C22 VIA
R12 - value passed in R2 when the device was claimed. This is usually a workspace pointer.

The device interrupt driver routine may corrupt R0-R3, R11 and R12. It should return using MOVS PC, R14. It is called in IRQ mode with interrupts disabled.

I/O_Podule_ClaimDevice (SWI &40501)

Purpose
To claim a User Port 65C22 VIA device interrupt.

Parameters
R0 - device number
R1 - address of device interrupt driver routine
R2 - value to be passed in R12 when the device interrupt driver routine is called

Results
R0-R2 preserved

Use
This call installs the device interrupt driver routine with the address specified in R1 on the device vector given in R0. If the same driver is already installed on the vector (ie the same parameters were used to install a previous driver) then the old copy is removed from the vector. Note that this call does not enable interrupts from the device. The previous driver is added to the list of earlier claimants.

I/O_Podule_ReleaseDevice (SWI &40502)

Purpose
To release a User Port 65C22 VIA device interrupt.

Parameters
R0 - device number
R1 - address of device interrupt driver routine
R2 - value to be passed in R12 when the device interrupt driver routine is called

Results
R0-R2 preserved

Use
This call removes the device interrupt driver routine with the address specified in R1 from the device vector given in R0. The driver is identified by the parameters which must be the same as when it was installed. The previous driver is re-installed at the head of the chain.

I/O_Podule_EnableDevice (SWI &40503)

Purpose
To enable interrupts from a User Port 65C22 VIA interrupt device by setting the appropriate bit in the Interrupt Enable Register (IER) of the 65C22 VIA.

Parameters
R0 - device number

Results
R0 preserved

I/O_Podule_DisableDevice (SWI &40504)

Purpose
To disable interrupts from a User Port 65C22 VIA interrupt device by clearing the appropriate bit in the Interrupt Enable Register (IER) of the 65C22 VIA.

Parameters
R0 - device number

Results
R0 preserved

I/O_Podule_GenerateADCEventOn (SWI &40505)

Purpose

To enable the calling of the OS_GenerateEvent SWI by the Analogue to Digital Converter interrupt handler. This is the default.

I/O_Podule_GenerateADCEventOff (SWI &40506)

Purpose

To disable the calling of the OS_GenerateEvent SWI by the Analogue to Digital Converter interrupt handler. This reduces the amount of time spent servicing the interrupt.

User Port 65C22 VIA Interrupt Example Program

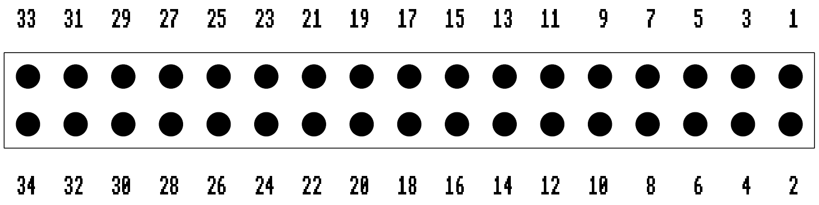
```
REM Area to assemble code
DIM code% 256

FOR pass%=0 TO 3 STEP 3

REM Address to assemble code
P%=code%

[
OPT pass%

.T1Interruptrt1
 MOV R0,#%01000000
 STRB R0,[R11,#52]
 LDR R0,[R12]
 ADD R0,R0,#1
 STR R0,[R12]
 MOVS R15,R14

.T1Interruptrt2
 MOV R0,#%01000000
 STRB R0,[R11,#52]
 LDR R0,[R12]
 SUB R0,R0,#1
 STR R0,[R12]
 MOVS R15,R14

.T1Interruptrt3
 MOV R0,#%01000000
 STRB R0,[R11,#52]
 LDR R0,[R12]
 EOR R0,R0,#&FF
 STR R0,[R12]
 MOVS R15,R14

; Variables

ALIGN
.Count1 EQUD 0
.Count2 EQUD 0
.Count3 EQUD 0
]
```

```
NEXTpass%

REM Configure 65C22 Timer 1 in Free-Running Mode

period% = &7FFF

REM Claim T1 Device T1Interruptrt1
SYS"I/O_Podule_ClaimDevice",4,T1Interruptrt1,Count1

REM write to T1L-L
SYS"OS_Byte",151,96+6,period%MOD256
REM write to T1L-H and clear T1 interrupt flag
SYS"OS_Byte",151,96+7,period%DIV256

REM update ACR
SYS"OS_Byte",150,96+11 TO ,,byte%
byte% = (byte% AND %00111111) OR %01000000
SYS"OS_Byte",151,96+11,byte%

REM write to T1C-H and transfer T1L-L to T1C-L
SYS"OS_Byte",151,96+5,period%DIV256

REM Turn off Generation of ADC EOC Event in I/O_Podule interrupt
handler
REM to reduce overall interrupt latency of computer
SYS"I/O_Podule_GenerateADCEventOff"

Count1!0 = 0

SYS"I/O_Podule_EnableDevice",4

FOR i% = 1 TO 3
 FOR j% = 1 TO 4
  PRINT "Count1 = ";Count1!0;" Channel ";j%;" Value (hex) "~ADVAL(j%)
 NEXT j%
NEXT i%

SYS"I/O_Podule_DisableDevice",4

REM Claim T1 Device T1Interruptrt2
SYS"I/O_Podule_ClaimDevice",4,T1Interruptrt2,Count2

REM Claim T1 Device T1Interruptrt3
SYS"I/O_Podule_ClaimDevice",4,T1Interruptrt3,Count3

REM Release T1 Device T1Interruptrt1
SYS"I/O_Podule_ReleaseDevice",4,T1Interruptrt1,Count1

Count3!0 = 0

SYS"I/O_Podule_EnableDevice",4

FOR i% = 1 TO 3
 FOR j% = 1 TO 4
  PRINT "Count3 = ";Count3!0;" Channel ";j%;" Value (hex) "~ADVAL(j%)
 NEXT j%
NEXT i%

SYS"I/O_Podule_DisableDevice",4

REM Release T1 Device T1Interruptrt3
SYS"I/O_Podule_ReleaseDevice",4,T1Interruptrt3,Count3

REM Turn on Generation of ADC EOC Event in I/O_Podule interrupt
handler
REM default power on state
SYS"I/O_Podule_GenerateADCEventOn"

Count2!0 = 0

SYS"I/O_Podule_EnableDevice",4

FOR i% = 1 TO 3
 FOR j% = 1 TO 4
  PRINT "Count2 = ";Count2!0;" Channel ";j%;" Value (hex) "~ADVAL(j%)
 NEXT j%
NEXT i%

SYS"I/O_Podule_DisableDevice",4

REM Release T1 Device T1Interruptrt2
SYS"I/O_Podule_ReleaseDevice",4,T1Interruptrt2,Count2

END
```

1MHz Bus

The 1MHz Bus provides two 256 byte pages of memory-mapped I/O address space. In the address space of the BBC Microcomputer they were decoded as NotPageFC (known as FRED) and NotPageFD (known as JIM).

The pin designations of the 1MHz Bus 34-way ribbon cable connector are shown below.



1 - 0V	2 - R/W	19 - D1	20 - D2
3 - 0V	4 - 1MHz	21 - D3	22 - D4
5 - 0V	6 - NMI	23 - D5	24 - D6
7 - 0V	8 - IRQ	25 - D7	26 - 0V
9 - 0V	10 - FC	27 - A0	28 - A1
11 - 0V	12 - FD	29 - A2	30 - A3
13 - 0V	14 - RST	31 - A4	32 - A5
15 - 0V	16 - NC	33 - A6	34 - A7
17 - 0V	18 - D0		

Notes

1. The page decode signal NotPageFC or NotPageFD is lo true when the ARM processor accesses the 1MHz Bus. They validate the 1MHz Bus address (A0..A7) and read/write (R/W) signals.

2. The least significant byte of the ARM processor 32-bit data bus is mapped to the 1MHz Bus data (D0..D7) bus.

3. As each byte is on a 32-bit word boundary the A2..A9 address lines of the ARM processor are mapped to the A0..A7 address lines of the 1MHz Bus.

4. The R/W signal specifies whether a read or write cycle is in progress.

5. The IRQ and NMI interrupt lines from the 1MHz Bus are not normally connected. However, they can be connected to either the IRQ or FIQ lines on the expansion backplane by option select links - see Appendix 2.

6. The analogue input to the sound channel of the BBC Microcomputer is not supported.

7. The 1MHz signal is the expansion backplane Ref8M signal divided down to produce the 1MHz Bus signal and the NotPageFC and NotPageFD signals are timed relative to it.

8. The RST signal is a buffered version of the expansion backplane RST signal.

1MHz Bus OS_Bytes (SWI &06)

OS_Byte 146

Purpose

To read a byte from NotPageFC.

Example

```
SYS"OS_Byte",146,offset% TO ,,byte%
```

Parameters

R0 - OS_Byte number 146
R1 - address offset in page

Results

R2 - byte read

OS_Byte 147

Purpose

To write a byte to NotPageFC.

Example

```
SYS"OS_Byte",147,offset%,byte%
```

Parameters

R0 - OS_Byte number 147
R1 - address offset in page
R2 - byte to be written

OS_Byte 148

Purpose

To read a byte from NotPageFD.

Example

```
SYS"OS_Byte",148,offset% TO ,,byte%
```

Parameters

R0 - OS_Byte number 148
R1 - address offset in page

Results

R2 - byte read

OS_Byte 149

Purpose

To write a byte to NotPageFD

Example

```
SYS"OS_Byte",149,offset%,byte%
```

Parameters

R0 - OS_Byte number 149
R1 - address offset in page
R2 - byte to be written

1MHz Bus Example Programs

Read a byte from Page FC

```
10 REM I/O Expansion Card
20 REM Read a byte from 1MHz Bus Page FC (FRED)
30 offset% = 128
40 SYS "OS_Byte",146,offset% TO ,byte%
50 PRINT "Byte read from 1MHz Bus Page FC (FRED)"
60 PRINT "Offset ";offset%;" Byte ";~byte%;" hex"
70 END
```

Write a byte to Page FD

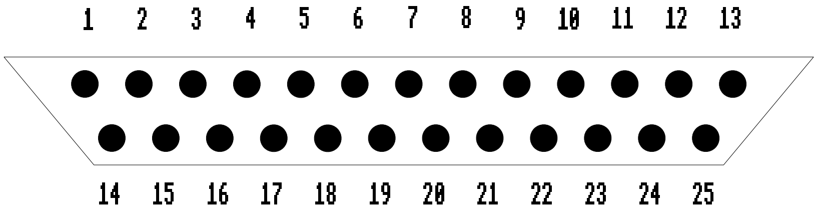
```
10 REM I/O Expansion Card
20 REM Write a byte to 1MHz Bus Page FD (JIM)
30 offset% = 64
40 byte% = &AA
50 SYS "OS_Byte",149,offset%,byte%
60 PRINT "Byte written to 1MHz Bus Page FD (JIM)"
70 PRINT "Offset ";offset%;" Byte ";~byte%;" hex"
80 END
```

User Port

The User Port consists of the eight data and two control lines from the port B of a 65C22 Versatile Interface Adapter (VIA). The 65C22 (VIA) is described in detail in Appendix 3.

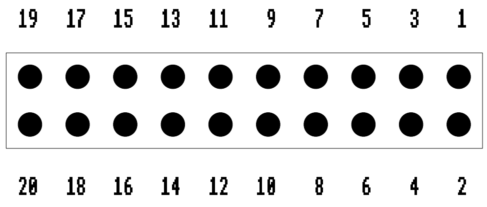
Pin Designations

25 pin Male D-Type Connector



1 - PB7	2 - PB6	11 - NC	12 - NC	21 - 0V
3 - PB5	4 - PB4	13 - NC	14 - 0V	22 - +5V
5 - PB3	6 - PB2	15 - 0V	16 - 0V	23 - +5V
7 - PB1	8 - PB0	17 - 0V	18 - 0V	24 - NC
9 - CB2	10 - CB1	19 - 0V	20 - 0V	25 - NC

20-way Ribbon Cable Connector (old design)



1 - +5V	2 - CB1	11 - 0V	12 - PB3
3 - +5V	4 - CB2	13 - 0V	14 - PB4
5 - 0V	6 - PB0	15 - 0V	16 - PB5
7 - 0V	8 - PB1	17 - 0V	18 - PB6
9 - 0V	10 - PB2	19 - 0V	20 - PB7

In the address space of the BBC Microcomputer the 16 registers of the 65C22 (VIA) were decoded at offsets &60 to &6F hex (96 to 111 decimal) in NotPageFE (known as SHEILA).

Notes

1. The 65C22 (VIA) on the I/O Expansion Card is clocked at 2MHz whereas the equivalent VIA in the BBC Microcomputer is clocked at 1MHz. This means that the interval timers run twice as fast.

2. The IRQ interrupt line from the 65C22 (VIA) is not normally connected. However, it can be connected to either the IRQ or FIQ lines on the expansion backplane by option select links. See Appendix 2.

User Port OS_Bytes (SWI &06)

OS_Byte 150

Purpose
To read a byte from the 65C22 VIA in NotPageFE

Example
```
SYS"OS_Byte",150,96+reg% TO ,,byte%
```

Parameters
R0 - OS_Byte number 150
R1 - address of offset in page of 65C22 VIA register &60 to &6F hex (96 to 111 decimal)

Results
R2 - byte read

OS_Byte 151

Purpose
To write a byte to the 65C2VIA in NotPageFE

Example
```
SYS"OS_byte",151,96+reg%,byte%
```

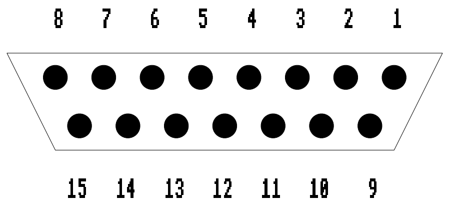
Parameters
R0 - OS_byte number 151
R1 - address of offset in page of 65C22 VIA register &60 to &6F hex (96 to 111 decimal)
R2 - byte to be written

User Port Example Program

```
 10 REM I/O Expansion Card
 20 REM Configure User Port data lines as outputs
 30 REM and output a data byte
 40 reg% = 2 :REM DDRB
 50 offset% = &60 + reg% :REM offset in Page FE (SHEILA)
 60 byte% = &FF :REM all data lines as outputs
 70 SYS "OS_Byte",151,offset%,byte%
 80 PRINT "User Port all data lines configured as outputs"
 90 reg% = 0 :REM ORB
100 offset% = &60 + reg% :REM offset in Page FE (SHEILA)
110 byte% = &A5 :REM data byte to output
120 SYS "OS_Byte",151,offset%,byte%
130 PRINT "User Port data byte output"
140 END
```

Analogue to Digital Converter

The same Analogue to Digital Converter IC as in the BBC Microcomputer is used. This is the uPD7002, a 10-bit integrating converter. However, in this implementation, only the most significant 8-bits can be relied upon. The pin designations of the ADC 15-way D-type connector on the rear panel are shown below.



1 - +5V	5 - AGND	9 - LPSTB	13 - FIRE0
2 - AGND	6 - AGND	10 - FIRE1	14 - VREF
3 - AGND	7 - CH1	11 - VREF	15 - CH0
4 - CH3	8 - AGND	12 - CH2	

Notes

1. The Analogue to Digital Converter will convert any input voltage in the range 0V - VRef (typically 0V to 1.8V).

Analogue to Digital Converter OS_Bytes (SWI &06)

OS_Byte 16

Purpose
To select the number of ADC channels to be converted.

Example
```
SYS"OS_Byte",16,chns%
```

Parameters
R0 - OS_byte number 16
R1 - number of ADC channels to be converted (0-4) (if 0 ADC is disabled)

OS_Byte 17

Purpose
To force the number of the next ADC channel to be converted.

Example
```
SYS"OS_Byte",17,chn%
```

Parameters
R0 - OS_Byte number 17
R1 - number of the next ADC channel to be converted

OS_Byte 128

Purpose
To read the ADC channel or fire button status

Parameters (to read the ADC channel)
R0 - OS_Byte number 128
R1 - channel number (1-4)

Results
R1 - least significant byte of 16-bit value
R2 - most significant byte of 16-bit value

Example
```
SYS"OS_Byte",128,chn% TO ,lsb%,msb%
```

Parameters (to read the fire button status)
R0 - OS_byte number 128
R1 - set to 0

Results
R1 - bits 0 and 1 indicate the status of the two fire buttons, bit 0 is the left fire button and bit 1 is the right fire button

R2 - channel number last used for an ADC conversion

Example
```
SYS"OS_Byte",128,0 TO ,buttons%,chn%
```

OS_Byte 188

Purpose
To read the current ADC channel

Example
```
SYS"OS_Byte",188 TO ,chn%
```

Parameters
R0 - OS_Byte number 188

Results
R1 - current ADC channel

OS_Byte 189

Purpose
To read the maximum ADC channel

Example
```
SYS"OS_Byte",189 TO ,maxchn%
```

Parameters
R0 - OS_byte number 189

Results
R1 - maximum ADC channel (set by OS_byte 16)

OS_Byte 190

Purpose
To read the precision of conversion

Example
SYS"OS_Byte",190 TO ,precision%

Parameters
R0 - OS_Byte number 190

Results
R1 - precision
    if R1 = 0 or 12 then the conversion is 12 bit
    if R1 = 8 then the conversion is 8 bit
Note that in this implementation only the most significant 8 bits can be relied upon.

Analogue to Digital Converter Example Programs

Using ADVAL

```
   10 REM I/O Expansion Card
   20 REM Analogue to Digital Converter
   30 REM Read value of channel 2
   40 REM Read fire button status and last channel used
   50 REM Using ADVAL
   60
   70 channel% = 2
   80 reading% = ADVAL(2)
   90 buttons% = ADVAL(0)
  100
  110 PRINT "ADC reading on ";channel%;" ";(1.8*reading%)/&FFFF;"
volts"
  120 PRINT "Fire button status ";buttons% AND %11 :REM bits 0 and 1
  130 PRINT "Last channel used ";buttons% >> 8 :REM bits 8 to 15
  140 END
```

Using OS_Bytes

```
   10 REM I/O Expansion Card
   20 REM Analogue to Digital Converter
   30 REM Read value of channel 2
   40 REM Read fire button status and last channel used
   50 REM Using OS_Byte calls
   60
   70 channel% = 2
   80 status% = 0
   90 SYS "OS_Byte",128,channel% TO ,lsbyte%,msbyte%
  100 SYS "OS_Byte",128,status% TO ,buttons%,last_channel%
  110 reading% = (msbyte% << 8) OR lsbyte%
  120
  130 PRINT "ADC reading on ";channel%;" ";(1.8*reading%)/&FFFF;"
volts"
  140 PRINT "Fire button status ";buttons% AND %11 :REM bits 0 and 1
  150 PRINT "Last channel used ";last_channel% :
  160 END
```

Appendix 1

I/O Expansion Card Memory Map

Slot	Device	Address space
0	1MHz Bus FRED	0300 0000 to 0300 03FF step 4 (256 bytes)
	1MHz Bus JIM	0300 0800 to 0300 0BFF step 4 (256 bytes)
	ADC	0300 1000 to 0300 100C step 4 (4 bytes)
	ROM IC2	033C 0000 to 033C 1FFF step 4 (2 Kbytes)
	User Port (VIA)	033C 2000 to 033C 203C step 4 (16 bytes)
	Interrupt Status	033C 3000 (1 byte)
1	1MHz Bus FRED	0300 4000 to 0300 43FF step 4 (256 bytes)
	1MHz Bus JIM	0300 4800 to 0300 4BFF step 4 (256 bytes)
	ADC	0300 5000 to 0300 500C step 4 (4 bytes)
	ROM IC2	033C 4000 to 033C 5FFF step 4 (2 Kbytes)
	User Port (VIA)	033C 6000 to 033C 603C step 4 (16 bytes)
	Interrupt Status	033C 7000 (1 byte)
2	1MHz Bus FRED	0300 8000 to 0300 83FF step 4 (256 bytes)
	1MHz Bus JIM	0300 8800 to 0300 8BFF step 4 (256 bytes)
	ADC	0300 9000 to 0300 900C step 4 (4 bytes)
	ROM IC2	033C 8000 to 033C 9FFF step 4 (2 Kbytes)
	User Port (VIA)	033C A000 to 033C A03C step 4 (16 bytes)
	Interrupt Status	033C B000 (1 byte)
3	1MHz Bus FRED	0300 C000 to 0300 C3FF step 4 (256 bytes)
	1MHz Bus JIM	0300 C800 to 0300 CBFF step 4 (256 bytes)
	ADC	0300 D000 to 0300 D00C step 4 (4 bytes)
	ROM IC2	033C C000 to 033C DFFF step 4 (2 Kbytes)
	User Port (VIA)	033C E000 to 033C E03C step 4 (16 bytes)
	Interrupt Status	033C F000 (1 byte)
4	1MHz Bus FRED	0300 0000 to 0300 03FF step 4 (256 bytes)
	1MHz Bus JIM	0300 0800 to 0300 0BFF step 4 (256 bytes)
	ADC	0300 1000 to 0300 100C step 4 (4 bytes)
	ROM IC2	033F 0000 to 033F 1FFF step 4 (2 Kbytes)
	User Port (VIA)	033F 2000 to 033F 203C step 4 (16 bytes)
	Interrupt Status	033F 3000 (1 byte)
5	1MHz Bus FRED	0300 4000 to 0300 43FF step 4 (256 bytes)
	1MHz Bus JIM	0300 4800 to 0300 4BFF step 4 (256 bytes)
	ADC	0300 5000 to 0300 500C step 4 (4 bytes)
	ROM IC2	033F 4000 to 033F 5FFF step 4 (2 Kbytes)
	User Port (VIA)	033F 6000 to 033F 603C step 4 (16 bytes)
	Interrupt Status	033F 7000 (1 byte)
6	1MHz Bus FRED	0300 8000 to 0300 83FF step 4 (256 bytes)
	1MHz Bus JIM	0300 8800 to 0300 8BFF step 4 (256 bytes)
	ADC	0300 9000 to 0300 900C step 4 (4 bytes)
	ROM IC2	033F 8000 to 033F 9FFF step 4 (2 Kbytes)
	User Port (VIA)	033F A000 to 033F A03C step 4 (16 bytes)
	Interrupt Status	033F B000 (1 byte)

7	1MHz Bus FRED	0300 C000 to 0300 C3FF step 4	(256 bytes)
	1MHz Bus JIM	0300 C800 to 0300 CBFF step 4	(256 bytes)
	ADC	0300 D000 to 0300 D00C step 4	(4 bytes)
	ROM IC2	033F C000 to 033F DFFF step 4	(2 Kbytes)
	User Port (VIA)	033F E000 to 033F E03C step 4	(16 bytes)
	Interrupt Status	033F F000	(1 byte)

Interrupt Status Bits

```
Bit 0  IRQ status (0 = no IRQ interrupt, 1 = IRQ interrupt)

Bit 2  FIQ status (0 = no FIQ interrupt, 1 = FIQ interrupt)
```

Appendix 2

Link Selection Options

S1	A-B	VIA interrupts connected to computer IRQ
	B-C	VIA interrupts connected to computer FIQ
S2	A-B	ADC interrupts connected to computer IRQ
	B-C	ADC interrupts connected to computer FIQ
S3		not used
S4	A-B	1MHz Bus IRQ interrupts connected to computer IRQ
	B-C	1MHz Bus IRQ interrupts connected to computer FIQ
S5	A-B	1MHz Bus NMI interrupts connected to computer IRQ
	B-C	1MHz Bus NMI interrupts connected to computer FIQ

Note that interrupts connected to the computer are enabled by setting the 65C22 VIA CA2 output low.

Appendix 3

The 65C22 Versatile Interface Adapter

The 65C22 Versatile Interface Adapter (VIA) provides

two 8-bit bi-directional peripheral data ports with input data latching

three bi-directional peripheral control lines

one input peripheral control line

two programmable 16-bit counter/timers

an 8-bit shift register for serial to parallel and parallel to serial conversion.

Control of peripheral devices is handled primarily through the two 8-bit bi-directional ports. Each of these lines can be programmed to act as either an input or an output. Several peripheral I/O lines can be controlled directly from the interval timers for generating programmable-frequency square waves and for counting externally generated pulses. To facilitate control of the chip, the internal registers have been organised into an interrupt flag register, an interrupt enable register and a pair of function control registers.

Registers

The 65C22 has 16 internal registers as shown in Table 1.

Peripheral Interface

This section contains a description of the peripheral data and control lines which are used to drive peripheral devices under control of the internal 65C22 registers. The operation of these peripheral lines is described in detail in subsequent sections.

Register Number	Register Designation	Description	
		Write	Read
0	ORB/IRB	Output Register "B"	Input Register "B"
1	ORA/IRA	Output Register "A"	Input Register "A"
2	DDRB	Data Direction Register "B"	
3	DDRA	Data Direction Register "A"	
4	T1C-L	T1 Low-Order Latches	T1 Low-Order Counter
5	T1C-H	T1 High-Order Counter	
6	T1L-L	T1 Low-Order Latches	
7	T1L-H	T1 High-Order Latches	
8	T2C-L	T2 Low-Order Latches	T2 Low-Order Counter
9	T2C-H	T2 High-Order Counter	
10	SR	Shift Register	
11	ACR	Auxiliary Control Register	
12	PCR	Peripheral Control Register	
13	IFR	Interrupt Flag Register	
14	IER	Interrupt Enable Register	
15	ORA/IRA	Same as Reg 1 Except No "Handshake"	

Table 1

Peripheral A Port (PA0 - PA7)
The Peripheral A port consists of eight lines which can be individually programmed to act as an input or an output under control of a Data Direction Register (DDRA). The level of output pins is controlled by an Output Register (ORA) and input data can be latched into an Input Register (IRA) under control of the CA1 line.

Peripheral A Control Lines (CA1, CA2)
The two Peripheral A port control lines act as interrupt inputs or as a handshake pair, one input and one output. Each line controls an internal interrupt flag with a corresponding interrupt enable bit. In addition, CA1 controls the latching of data on Peripheral A port input lines. The various modes of operation are controlled by the system processor through the internal control registers.

Peripheral B Port (PB0 - PB7)
The Peripheral B port consists of eight bi-directional lines which are controlled by an Output Register (ORB) and a Data Direction Register (DDRB) in the same manner as the Peripheral A port. The polarity of the PB7 output signal can be controlled by one of the interval timers while the second timer can be programmed to count pulses on the PB6 pin.

Peripheral B Control Lines (CB1, CB2)
The Peripheral B port control lines act as interrupt inputs or as a handshake pair, one input and one output. As with CA1 and CA2, each line controls an interrupt flag with a corresponding interrupt enable bit. In addition, these lines act as a serial port under control of the Shift Register.

Port A Registers, Port B Registers

Three registers are used in accessing each of the 8-bit peripheral ports. Each port has a Data Direction Register (DDRA, DDRB) for specifying whether the peripheral pins are to act as inputs or outputs. A "0" in a bit of the Data Direction Register causes the corresponding peripheral pin to act as an input. A "1" causes the pin to act as an output.

When the pin is programmed to act as an output, the voltage on the pin is controlled by the corresponding bit of the Output Register (ORA, ORB). A "1" in the Output Register causes the pin to go high, and a "0" causes the pin to go low. Data written into Output Register bits corresponding to pins programmed to act as inputs will be unaffected.

Reading a peripheral port causes the contents of the Input Register (IRA, IRB) to be transferred onto the Data Bus. With input latching disabled, IRA will always reflect the data on the PA pins. With input latching enabled (ACR, bit 0), setting the CA1 Interrupt Flag (IFR1) by an active transition on CA1, will cause IRA to latch the contents of the PA pins until the Interrupt Flag is cleared.

The IRB register operates in a similar manner. However, for output pins, the corresponding IRB bit will reflect the contents of the Output Register bit instead of the actual pin. This allows proper data to be read into the processor if the output pin is not allowed to go to full high voltage eg driving transistors. If input latching is enabled on Port B, setting the CB1 Interrupt Flag will cause IRB to latch this combination of input data and ORB data until the Interrupt Flag is cleared.

Handshake Control
The 65C22 allows positive control of data transfers between the system processor and peripheral devices through the operation of "handshake" lines. Port A lines (CA1, CA2) handshake data on both a read and a write operation while the Port B lines (CB1, CB2) handshake on a write operation only.

Read Handshake

Positive control of data transfers from peripheral devices into the system processor can be accomplished effectively using "Read" handshaking. In this case, the peripheral device must generate "Data Ready" to signal the processor that valid data is present on the peripheral port. This signal normally interrupts the processor, which then reads the data, causing generation of a "Data Taken" signal. The peripheral device responds by making new data available. This process continues until the data transfer is complete.

In the 65C22, automatic "Read" handshaking is possible on the PA port only. The CA1 interrupt pin accepts the "Data Ready" signal and CA2 generates the "Data Taken" signal. The "Data Ready" signal will set an internal flag which may interrupt the processor or which can be polled under software control. The "Data Taken" signal can be either a pulse or a DC level which is set low by the system processor and is cleared by the" Data Ready" signal.

Write Handshake

The sequence of operations which allows handshaking data from the system processor to a peripheral device is very similar to that described for Read Handshaking. However, for "Write" handshaking, the processor must generate the "Data Ready" signal (through the 65C22) and the peripheral device must respond with the "Data Taken" signal. This can be accomplished on both the PA port and the PB port on the 65C22. CA2 or CB2 acts as a "Data Ready" output in either the DC level or pulse mode and CA1 or CB1 accepts the "Data Taken" signal from the peripheral device, setting the interrupt flag and clearing the "Data Ready" output.

Timer 1

Interval Timer T1 consists of two 8-bit latches and a 16-bit counter. The latches are used to store data which is to be loaded into the counter. After loading, the counter decrements at system clock rate (2 MHz). Upon reaching zero, an interrupt flag will be set, and IRQ will go low if enabled. The timer will then disable any further interrupts, or will automatically transfer the contents of the latches into the counter and will continue to decrement. In addition, the timer can be instructed to invert the output signal on peripheral pin PB7 each time it "times-out". Each of these modes is discussed separately below.

Writing The Timer 1 Registers

The operations which take place when writing to each of the four Timer 1 addresses are as shown in Table 2.

Register Number	Operation
4	Write into low-order latch
5	Write into high-order latch Write into high-order counter Transfer low-order latch into low order counter. Reset T1 interrupt flag (IFR6)
6	Write low-order latch
7	Write high-order latch. Reset T1 interrupt flag (IFR6)

Table 2

Note that the processor does not write directly into the low-order counter (T1C-L). Instead, this half of the counter is loaded automatically from the low-order latch when the processor writes into the high-order counter. In fact, it may not be necessary to write to the low-order counter in some applications since the timing operation is triggered by writing to the high-order counter. The second set of addresses allows the the processor to write into the latch register without affecting the count-down in progress. This is discussed in detail below.

Reading the Timer 1 Registers
For reading the Timer 1 registers, the four addresses relate directly to the four registers as shown in Table 3.

Register Number	Operation
4	Read T1 low-order counter. Reset T1 interrupt flag (IFR6)
5	Read T1 high-order counter
6	Read T1 low-order latch
7	Read T1 high-order latch

Table 3

Timer 1 Operating Modes

Two bits are provided in the Auxiliary Control Register (ACR) to allow selection of the T1 operating modes. These bits and the four possible modes are as shown in Table 4.

Timer 1 One-Shot Mode
The interval timer one-shot mode allows generation of a single interrupt for each timer load operation. As with any interval time, the delay between the "write T1C-H" operation and generation of the processor interrupt is a direct function of the data loaded into the timing counter. In addition to generating a single interrupt, Timer 1 can be programmed to produce a single negative pulse on the PB7 peripheral pin. With the output enabled (ACR7=1) a "write T1C-H" operation will cause PB7 to go low. PB7 will return high when Timer 1 times out. The result is a single programmable width pulse.

ACR7 Output Enable	ACR6 "Free-Run Enable	Mode
0	0	Generates a single time-out interrupt each time T1 is loaded. PB7 is disabled.
0	1	Generates continuous interrupts. PB7 is disabled.
1	0	Generate a single interrupt and an output pulse on PB7 for each T1 load operation.
1	1	Generate continuous interrupts and a square wave output on PB7

Table 4

Note that the PB7 output enable function will over-ride bit 7 of the Data Direction Register B. PB7 will act as an output if DDRB7=1 or if ACR7=1.

In the one-shot mode, writing into the high-order latch has no effect on the operation of Timer 1. However, it will be necessary to ensure that the low-order latch contains the proper data before initiating the countdown with a "write T1C-H" operation. When the processor writes into the high-order counter, T1L-H will also copy the data, the T1 interrupt flag will be cleared, the contents of the low-order latch will be transferred into the low-order counter, and the timer will begin to decrement at system clock rate. If the PB7 output is enabled, this signal will go low on the phase two following the write operation. When the counter reaches zero, the T1 interrupt flag will be set, the IRQ pin will go low (interrupt enabled), and the signal on PB7 will go high.

At this time the counter will continue to decrement at system clock rate. This allows the system processor to read the contents of the counter to determine the time since interrupt. However, the T1 interrupt flag cannot be set again unless a "write TIC-H"
operation has taken place.

Timer 1 Free-Running Mode
The most important advantage associated with the latches in T1 is the ability to produce a continuous series of evenly spaced interrupts and the ability to produce a square wave on PB7 whose frequency is not affected by variations in the processor response time. This is accomplished in the "free-running" mode.

In the free-running mode (ACR6=1), the interrupt flag is set and the signal on PB7 is inverted each time the counter reaches zero. However, instead of continuing to decrement from zero after a time-out, the timer automatically transfers the contents of the latch into the counter (16 bits) and continues to decrement from there. The interrupt flag can be cleared by writing TIC-H, by reading TIC-L or by writing directly into the flag as described below. However, it is not necessary to rewrite the timer to enable setting the interrupt flag on the next time-out.

Rewriting the counter will always re-initialise the time-out period. In fact, the time-out can be prevented completely if the processor continues to rewrite the timer before it reaches zero. Timer 1 will operate in this manner if the processor writes into the high-order counter (TIC-H). However, by loading the latches only, the processor can access the timer during each counting-down operation without affecting the time-out in progress. Instead, the data loaded into the latches will determine the length of the next time-out period. This capability is particularly valuable in the free-running mode with the output enabled. In this mode, the signal on PB7 is inverted and the interrupt flag is set with each time-out. By responding to the interrupts with the new data for the latches, the processor can determine the period of the next half cycle during each half cycle of the output signal on PB7. In this manner, very complex pulse width modulated waveforms can be generated.

Timer 2

Timer 2 operates as an interval timer (in the "one-shot" mode only), or as a counter for counting negative pulses on the PB6 peripheral pin. A single control bit is provided in the Auxiliary Control Register (ACR) to select between these two modes. This timer is comprised of a "write-only" low-order latch (T2L-L), a "read-only" low-order counter and a read/write high-order counter. The counter registers act as a 16-bit counter which decrements at system clock rate (2 MHz).Timer 2 addressing can be summarised as in Table 5.

Register Number	Write	Read
8	Write T2L-L	Read T2C-L. Clear Interrupt Flag
9	Write T2C-H. Transfer T2L-L to T2C-L Clear Interrupt Flag	Read T2C-H

Table 5

Timer 2 Interval Timer Mode
As an interval timer, T2 operates in the "one-shot" mode similar to Timer 1. In this mode, T2 provides a single interrupt for each "write T2C-H" operation. After timing out,
the counter will continue to decrement. However, setting of the interrupt flag will be disabled after initial time-out so that it will not be set by the counter continuing to decrement through zero. The processor must rewrite T2C-H to enable setting of the interrupt flag. The interrupt flag is cleared by reading T2C-L or by writing T2C-H.

Timer 2 Pulse Counting Mode
In the pulse counting mode, T2 serves primarily to count a pre-determined number of negative-going pulses on PB6. This is accomplished by first loading a number into T2. Writing into T2C-H clears the interrupt flag and allows the counter to decrement each time a pulse is applied to PB6. The interrupt flag will be set when T2 reaches zero. At this time the counter will continue to decrement with each pulse on PB6.

However, it is necessary to rewrite T2C-H to allow the interrupt flag to set on subsequent down-counting operations.

Shift Register

The Shift Register (SR) performs serial data transfers into and out of the CB2 pin under control of an internal modulo-8 counter. Shift pulses can be applied to the CB1 pin from an external source or, with the proper mode selection, shift pulses generated internally will appear on the CB1 pin for controlling shifting in external devices.

The control bits which allow control of the various shift register operating modes are located in the Auxiliary Control Register (ACR). These bits can be set and cleared by the system processor to select one of the operating modes discussed in the following paragraphs.

Shift Register Input Modes
Auxiliary Control Register ACR4 selects the shift register input or output mode. There are three input modes and four output modes, differing primarily in the source of the pulses which control the shifting operation. With ACR4=0 the input modes are selected by ACR3 and ACR2 as shown in Table 6.

ACR4	ACR3	ACR2	
0	0	0	Shift Register Disabled
0	0	1	Shift in under Control of Timer 2
0	1	0	Shift in at system clock rate
0	1	1	Shift in under Control of External Input Pulses

Table 6

All Shift Register inputs are sampled into the Shift Register during the system clock low immediately following the detection of the shift clock rising transition. This detection occurs during system clock high.

Mode 000 - Shift Register Disabled
The 000 mode is used to disable the Shift Register. In this mode the microprocessor can write or read the SR, but the shifting operation is disabled and operation of CB1 and CB2 is controlled by the appropriate bits in the Peripheral Control Register (PCR). In this mode the SR Interrupt Flag is disabled (held to a logic 0).

Mode 001 - Shift in Under Control of Timer 2
In this mode the shifting rate is controlled by the low-order eight bits of T2. Shift pulses are generated on the CB1 pin to control shifting in external devices. The time between transitions of this output clock is a function of the system clock (2 MHZ) period and the contents of the low-order T2 latch.

The shifting operation is triggered by writing or reading the Shift Register. Data are shifted first into the low-order bit of SR and are then shifted into the next-higher-order bit of the Shift Register on the trailing edge of each clock pulse. The input data should change on the negative edge of the clock pulse. These data are loaded into the Shift Register during the system clock cycle following the positive edge of the clock pulse. After eight clock pulses, the Shift Register Interrupt Flag will be set.

Mode 010 - Shift in at System Clock Rate
In this mode the shift rate is a direct function of the system clock frequency (2 MHz). CB1 becomes an output which generates shift pulses for controlling external devices. The shifting operation is triggered by reading or writing the Shift Register. Data is shifted first into bit 0 and are then shifted into the next-higher-order bit of the Shift Register on the positive edge of each clock pulse. After nine clock pulses, the Shift Register Interrupt Flag will be set, and the output clock pulses on CB1 will stop.

Mode 011 - Shift in Under Control of External Source
In this mode CB1 becomes an input. This allows an external device to load the shift register at its own pace. The shift register counter will interrupt the processor each time 8 bits have been shifted in. However, the shift register counter does not stop the shifting operation; it acts simply as a pulse counter. Reading or writing the Shift Register resets the Interrupt Flag and initialises the SR counter to count another eight pulses.

Note that data is shifted during the first system clock cycle following the positive edge of the CB1 shift pulse. For this reason, data must be held stable during the first full cycle following CB1 going high.

Shift Register Output Modes

The four shift register output modes are selected by setting the input/output control bit (ACR4) to a logic 1 and then selecting the specific output mode with ACR 3 and ACR2. In each of these modes the shift register shifts data out of bit 7 to the CB2 pin. At the same time the contents of bit 7 are shifted back into bit 0. As in the input modes, CB1 is used either as an output to provide shifting pulses or as an input to allow shifting from an external pulse. The four modes are as shown in Table 7.

ACR4	ACR3	ACR3	Mode
1	0	0	Shift Out - Free-Running Mode. Shift Rate Controlled by T2
1	0	1	Shift Out - Shift Rate Controlled by T2
1	1	0	Shift Out at System Clock Rate
1	1	1	Shift Out Under Control of an External Pulse

Table 7

All shift register outputs are set during system clock low immediately following the detection of the shift clock falling transitions. This occurs during system clock high.

Mode 100 - Free-Running Output
This mode is very similar to mode 101 in which the shifting rate is set by T2. However, in mode 100 the SR Counter does not stop the shifting operation. Since the Shift Register bit 7 (SR7) is recirculated back into bit 0, the eight bits loaded into the Shift Register will be clocked onto CB2 repetitively. In this mode the shift register counter is disabled.

Mode 101 - Shift Out Under Control of Timer 2
In this mode the shift rate is controlled by Timer 2. However, with each read or write of the Shift Register the SR Counter is reset and 8 bits are shifted onto CB2. At the same time, eight shift pulses are generated on CB1 to control shifting in external devices. After the eight shift pulses, the shifting is disabled, and the SR Interrupt Flag is set. If the Shift Register is reloaded before the last time-out, the shifting will continue.

Mode 110 - Shifting Out at System Clock Rate
In this mode the shift register operation is similar to that of mode 101. However, the shifting rate is a function of the system clock rate and is independent of T2. Timer 2 resumes its normal function as an independent interval timer.

Mode 111 - Shift Out Under Control of an External Pulse
In this mode, shifting is controlled by pulses applied to the CB1 pin by an external device. The SR counter sets the SR Interrupt flag each time it counts eight pulses but it does not disable the shifting function. Each time the system processor writes or reads the shift register, the SR Interrupt flag is reset and the SR Counter is initialised to begin counting the next eight shift pulses on pin CB1. After eight shift pulses, the interrupt flag is set. The system processor can then load the shift register with the next byte of data.

Interrupt Control

Controlling interrupts within the 65C22 involves three principal operations; flagging the interrupts, enabling interrupts and signalling to the processor that an active interrupt exists within the chip. Interrupt flags are set by interrupting conditions which exist within the chip or on inputs to the chip. These flags normally remain set until the
interrupt has been serviced. To determine the source of an interrupt, the system processor must examine these flags in order from highest to lowest priority.

Associated with each interrupt flag is an interrupt enable bit. This bit can be set or cleared by the processor to enable interrupting the processor from the corresponding interrupt flag. If an interrupt flag is set to a logic 1 by an interrupting condition, and the corresponding interrupt enable bit is set to a 1, the Interrupt Request Output (IRQ) will go low interrupting the system processor. In the 65C22, all the interrupt flags are contained in one register. In addition, bit 7 of this register will be read as a logic 1 when an interrupt exists within the chip. This permits very convenient polling of several devices within a system to locate the source of an interrupt.

	7	6	5	4	3	2	1	0
Interrupt Flag Register	IRQ	T1	T2	CB1	CB2	SR	CA1	CA2
Interrupt Enable Register	Set/ clear control	T1	T2	CB1	CB2	SR	CA1	CA2

Table 8

Interrupt Flag Register (IFR)

Bit 7 of the IFR indicates the status of the IRQ output. This bit corresponds to the logic function : IRQ = IFR6 x IER6 + IFR5 x IER5 + IFR4 x IER4 + IFR3 x IER3 + IFR2 x IER2 + IFR1 x IER1 + IFR0 x IER0.

Note: x = logic AND, + = Logic OR.

Bits six through zero are latches which are set and cleared as shown in Table 9.

In addition to the clearing operations shown in the table, individual bits in the IFR can be cleared by writing into the register. A logic 1 in the data word written into the IFR will clear the corresponding interrupt flag. A zero in this word will leave the corresponding flag untouched. Setting the flag occurs only from interrupting conditions within the chip.

Bit	Set by	Cleared by
0	Active transition of the signal on the CA2 pin	Reading or writing the A Port Output Register (ORA) using address 0001
1	Active transition of the signal on the CA1 pin	Reading or writing the A Port Output Register (ORA) using address 0001
2	Completion of eight shifts	Reading or writing the Shift Register
3	Active transition of the signal on the CB2 pin	Reading or writing the B Port Output Register
4	Active transition of the signal on the CB1 pin	Reading or writing the B Port Output Register
5	Time-out of Timer 2	Reading T2 low order counter or writing T2 high order counter
6	Time-out of Timer 1	Reading T1 low order counter or writing T1 high order latch

Table 9

The IFR bit 7 is not a flag. Therefore, this bit is not directly cleared by writing a logic 1 into it. It can only be cleared by clearing all the flags in the register or by disabling all the active interrupts as discussed in the next section.

Interrupt Enable Register (IER)

For each interrupt flag in the IFR, there is a corresponding bit in the IER. The system processor can set or clear selected bits in this register to facilitate controlling individual interrupts without affecting others.

If bit 7 is 0 during a write operation, each 1 in bits 6 through 0 clears the corresponding bit in the IER. For each 0 in bits 6 through 0 the corresponding bit is unaffected. If bit 7 is a 1 during a write operation each 1 in bits 6 through 0 sets the corresponding bit in the IER. For each 0 in bits 6 through 0 the corresponding bit is unaffected.

In addition to setting and clearing IER bits, the processor can read the contents of this register. Bit 7 will be read as a logic 0.

Function Control

Control of the various functions and operating modes within the 65C22 is accomplished primarily through two register, the Peripheral Control Register (PCR), and the Auxiliary Control Register (ACR). The PCR is used primarily to select the operating mode for the four peripheral control pins. The ACR selects the operating mode for the interval timers (T1, T2) and the serial port (SR).

Peripheral Control Register

The Peripheral Control Register is organised as shown in Table 10. Each of these functions is discussed in detail below.

CA1 Control
Bit 0 of the PCR selects the active transition of the input signal applied to the CA1 interrupt input pin.. If this bit is a logic 0, the CA1 interrupt flag will be set by a negative transition (high to low) of the signal on the CA1 pin. IF PCR0 is a logic 1, the CA1 interrupt flag will be set by a positive transition (low to high) of this signal.

Bit No	7	6	5	4	3	2	1	0
Function		CB2 Control		CB1 Control		CA2 Control		CA1 Control

Table 10

CA2 Control
The CA2 pin can be programmed to act as an interrupt input or as a peripheral control output. As an input, CA2 operates in two modes, differing primarily in the methods available for resetting the interrupt flag. Each of these two input modes can operate with either a positive or a negative active transition as described above for CA1.

In the output mode, the CA2 will perform either a "Read" or a "write" handshake operation. The CA2 operating modes are selected as shown in Table 11.

PCR3	PCR2	PCR1	Mode
0	0	0	CA2 Negative Edge Interrupt (IFR0/ORA Clear) Mode -- Set CA2 interrupt flag (IFR0) on a negative transition of the input signal. Clear IFR0 on a read or write of the Peripheral A Output Register (ORA) or by writing logic 1 into IFR0.
0	0	1	CA2 Negative Edge Interrupt (IFR0 Clear) Mode -- Set IFR0 on a negative transition of the CA2 input signal. Reading or writing ORA does not clear the CA2 interrupt flag. Clear IFR0 by writing logic 1 into IFR0.
0	1	0	CA2 Positive Edge Interrupt (IFR0/ORA Clear) Mode -- Set CA2 interrupt flag on a positive transition of the CA2 input signal. Clear IFR0 with a read or write of the Peripheral A Output Register.
0	1	1	CA2 Positive Edge Interrupt (IFR Clear) Mode -- Set IFR0 on a positive transition of the CA2 input signal. Reading or writing ORA does not clear the CA2 interrupt flag. Clear IFR0 by writing logic 1 into IFR0.
1	0	0	CA2 Handshake Output Mode -- Set CA2 output low on a read or write of the Peripheral A Output Register. Reset CA2 high with an active transition on CA1.
1	0	1	CA2 Pulse Output Mode -- CA2 goes low for one cycle following a read or write of the Peripheral A Output Register.
1	1	0	CA2 Output Low Mode -- The CA2 output is held low in this mode.
1	1	1	CA2 Output High Mode -- The CA2 output is held high in this mode.

Table 11

In the interrupt (IFR0 clear) input mode, writing or reading the ORA register has no effect on the CA2 interrupt flag. This flag must be cleared by writing a logic 1 into the appropriate IFR bit. This mode allows the processor to handle interrupts which are independent of any operations taking place on the peripheral data ports.

The handshake and pulse output modes have been described previously. Note that the timing of the output signal varies slightly depending on whether the operation is initiated by a read or a write.

CB1 Control
Control of the active transition of the CB1 input signal operates in exactly the same manner as that described above for CA1. If PCR4 is a logic 1, the CB1 interrupt flag (IFR4) is a logic 1, the CB1 interrupt flag (IFR4) will be set by a negative transition of the CB1 input signal and cleared by a read or write of the ORB register. If PCR4 is a logic one, IFR4 will be set by a positive transition of CB1.

If the Shift Register function has been enabled, CB1 will act as an input or output for the shift register clock signals. In this mode, the CB1 interrupt flag will still respond to the selected transition of the signal on the CB1 pin.

CB2 Control
With the serial port disabled, operation of the CB2 pin is a function of the three high-order bits of the PCR. The CB2 modes are very similar to those described previously for CA2. These modes are selected as shown in Table 12.

PCR7	PCR6	PCR5	Mode
0	0	0	CB2 Negative Edge Interrupt (IFR3/ORB Clear) Mode -- Set CB2 interrupt flag (IFR3) on a negative transition of the CB2 input signal. Clear IFR3 on a read or write of the Peripheral B Output Register (ORB) or by writing a logic 1 into IFR3.
0	0	1	CB2 Negative Edge Interrupt (IFR3 Clear) Mode -- Set IFR3 on a negative transition of the CB2 input signal. Reading or writing ORB does not clear the interrupt flag. Clear IFR3 by writing logic 1 into IFR3.
0	1	0	CB2 Positive Edge Interrupt (IFR3/ORB Clear) Mode -- Set CB2 input signal. Clear the CB2 interrupt flag on a read or write of the ORB or by writing logic 1 into IFR3.
0	1	1	CB2 Positive Edge Interrupt (IFR3 Clear) Mode -- Set IFR3 on a positive transition of the CB2 input signal. Reading or writing ORB does not clear the CB2 interrupt flag. Clear IFR3 by writing logic 1 into IFR3.
1	0	0	CB2 Handshake Output Mode -- Set CB2 low on a write ORB operation. Reset CB2 high with an active transition of the CB1 input signal.
1	0	1	CB2 Pulse Output Mode -- Set CB2 low for one cycle following a write ORB operation
1	1	0	CB2 Manual Output Low Mode -- The CB2 output is held low in this mode.
1	1	1	CB2 Manual Output High Mode -- The CB2 output is held high in this mode.

Table 12

Auxiliary Control Register

Many of the functions in the ACR have been discussed previously. However, a summary of this register is presented here as a convenient reference. The ACR is organised as shown in Table 13.

Bit No	7	6	5	4	3	2	1	0
Function	T1 Control	T2		Shift Register		PB	PA	
			Control		Control		Latch Enable	Latch Enable

Table 13

PA Latch Enable
The 65C22 provides input latching on both the PA and PB ports. In this mode, the data present on the peripheral A input pins will be latched within the chip when the CA1 interrupt flag is set. Reading the PA port will result in these latches being transferred into the system processor. As long as the CA1 interrupt flag is set, the data on the peripheral pins can change without affecting the data in the latches. This input latching can be used with any of the CA2 input or output modes.

It is important to note that on the PA port, the system processor always reads the data on the peripheral pins (as reflected in the latches). For output pins, the processor still reads the latches. This may or may not reflect the data currently in the ORA. Proper system operation requires careful planning on the part of the system designer if input latching is combined with output pins on the peripheral ports.

Input latching is enabled by setting bit 0 in the ACR to a logic 1. As long as this bit is a 0, the latches will directly reflect the data on the pins.

PB Latch Enable
Input latching on the PB port is controlled in the same manner as that described for the PA port. However, with the PB port the input latch will store either the voltage on the pin or the contents of the Output Register (ORB) depending on whether the pin is programmed to act as an input or an output. As with the PA port, the system processor always reads the input latches.

Shift Register Control
The Shift Register operating mode is selected as shown in Table 14.

ACR4	ACR3	ACR2	Mode
0	0	0	Shift register disabled
0	0	1	Shift in under control of Timer 2
0	1	0	Shift in under control of f2 pulses
0	1	1	Shift in under control of external clock pulses
1	0	0	Free-running output at rate determined by Timer 2
1	0	1	Shift out under control of Timer 2
1	1	0	Shift out under control of the o2 pulses
1	1	1	Shift out under control of external clock pulses

Table 14

T2 Control
Timer 2 operates in two modes. If ACR5 = 0, T2 acts as an interval timer in the one-shot mode. If ACR5 = 1, Timer 2 acts to count a predetermined number of pulses on pin PB6.

T1 Control
Timer 1 operates in the one-shot or free-running mode with the PB7 output control enabled or disabled. These modes are selected as shown in Table 15.

ACR7	ACR6	Mode
0	0	One-Shot Mode -- Output to PB7 disabled
0	1	Free-Running Mode -- Output to PB7 disabled
1	0	One-Shot Mode -- Output to PB7 enabled
1	1	Free-Running Mode -- Output to PB7 enabled

Table 15